Bluetooth

Bluetooth is a communications protocol designed for short-range, low-bandwidth peer-to-peer communications.

**BluetoothAdapter** The Bluetooth Adapter represents the local Bluetooth device.

**BluetoothDevice** Each remote device with which you wish to communicate is represented as a BluetoothDevice.

**BluetoothSocket** Call `createRfcommSocketToServiceRecord` on a remote Bluetooth Device object to create a Bluetooth Socket that will let you make a connection request to the remote device, and then initiate communications.

**BluetoothServerSocket** By creating a Bluetooth Server Socket (using the `listenUsingRfcommWithServiceRecord` method) on your local Bluetooth Adapter, you can listen for incoming connection requests from Bluetooth Sockets on remote devices.

Use the `isEnabled` method, as shown in Listing 13-2, to confirm the device is enabled before accessing these properties. If not enabled all methods will return null!
BluetoothAdapter bluetooth = BluetoothAdapter.getDefaultAdapter();
String toastText;
if (bluetooth.isEnabled()) {
    String address = bluetooth.getAddress();
    String name = bluetooth.getName();
    toastText = name + " : " + address;
} else
    toastText = "Bluetooth is not enabled";

Toast.makeText(this, toastText, Toast.LENGTH_LONG).show();

**Needed permissions**

<uses-permission android:name="android.permission.BLUETOOTH"/>
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>

**Enabling Bluetooth and being Discoverable**

Straight forward, you might want to know the result of this using `onActivityResult` method

String enableBT = BluetoothAdapter.ACTION_REQUEST_ENABLE;
startActivityForResult(new Intent(enableBT), 0);

String aDiscoverable = BluetoothAdapter.ACTION_REQUEST_DISCOVERABLE;
startActivityForResult(new Intent(aDiscoverable), 1);

you can use the second one only and the device will turn on Bluetooth automatically ! :D
Searching for devices

We will use receivers to get founded devices info ,, but we will use both receivers and filters in code ,, same idea but using code:

Implement the receiver

BroadcastReceiver discoveryResult = new BroadcastReceiver() {
    public void onReceive(Context context, Intent intent) {
        String remoteDeviceName = intent.getStringExtra(BluetoothDevice.EXTRA_NAME);
        BluetoothDevice remoteDevice = intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
        Toast.makeText(getApplicationContext(), "Discovered: " + remoteDeviceName, Toast.LENGTH_SHORT).show();
        // TODO Do something with the remote Bluetooth Device.
    }
};

Register your receiver with the filter

registerReceiver(discoveryResult, new IntentFilter(BluetoothDevice.ACTION_FOUND));

Start Scanning

if (!bluetooth.isDiscovering())
    bluetooth.startDiscovery();

Bluetooth Communications

You can establish an RFCOMM communication channel for bidirectional communications using the following classes.

➤ BluetoothServerSocket Used to establish a listening socket for initiating a link between devices. To establish a handshake, one device acts as a server to listen for, and accept, incoming connection requests.
BluetoothSocket Used in creating a new client socket to connect to a listening Bluetooth Server Socket, and returned by the Server Socket once a connection is established. Once the connection is made, Bluetooth Sockets are used on both the server and client sides to transfer data streams.

Codes:

Server

```java
UUID uuid = UUID.fromString("a60f35f0-b93a-11de-8a39-08002009c666");
String name = "bluetoothserver";

try {
    btserver = bluetooth.listenUsingRfcommWithServiceRecord(name, uuid);

    Thread acceptThread = new Thread(new Runnable() {
        public void run() {
            try {
                // Block until client connection established.
                BluetoothSocket serverSocket = btserver.accept();

                // TODO Transfer data using the server socket

                Toast.makeText(getApplicationContext(), "new dev connected: ",
                Toast.LENGTH_SHORT).show();

                } catch (IOException e) {
                    Log.d("BLUETOOTH", e.getMessage());
                }
            }
        }
    });

    acceptThread.start();

} catch (IOException e1) {
    // TODO Auto-generated catch block
    e1.printStackTrace();
}
```
Client

```java
try {
    BluetoothDevice device = bluetooth.getRemoteDevice("00:23:76:35:2F:AA");
    BluetoothSocket clientSocket = device.createRfcommSocketToServiceRecord(uuid "same as server");
    clientSocket.connect(); // blocking operation
}
```
message = message + new String(buffer, 0, bytesRead - 1);
return message;
}
}

} catch (IOException e) {
}
return message;